Middle-East Journal of Scientific Research 19 (12): 1623-1629, 2014
ISSN 1990-9233
© IDOSI Publications, 2014
DOI: 10.5829/idosi.mejsr.2014.19.12.11265

Data-Acquisition System Using Embedded Linux

S. *Philomina*

Department of ECE, Bharath University, Chennai, India

Abstract: In this paper, we present the principles of a low-operational-cost but flexible Internet-based data-acquisition system. The main core of the system is an embedded hardware running a scaled-down version of Linux: a popular choice of operating system for embedded applications. The embedded device communicates through General Packet Radio Service (GPRS), which makes it accessible from anywhere in the world through a web server built into the embedded device. In addition, GPRS provides a bidirectional real-time data transfer allowing interaction. The proposed system eliminates the need for server software and maintenance. A novel approach is introduced to minimize the operational costs while operating with a large amount of data. The system is demonstrated to be suitable for different embed- ded applications by attaching several real-time modules through appropriate interfaces.

Key words: Data-acquisition · Embedded system · Interaction · Internet · Linux · Real time

INTRODUCTION

DATA-ACQUISITION systems with remote accessibility are in great demand in industry and consumer applications. In some applications, human beings have been replaced by unmanned devices that will acquire data and relay the data back to the base [1]. There are data-acquisition and control devices that will be a substitute for a supervisor in a multisite job operation. A single person can monitor and even interact with the ongoing work from a single base station. An acquisition unit designed to collect data in their simplest form is detailed in [2], which is based on Linux [3], which is a popular choice for embedded PC systems. A similar system in [4] provides data-acquisition with no concern for remote access. Data collection for post processing on a vehicle's position for an advanced traffic survey is discussed in [5].

Some applications adding remote accessibility are detailed in [6] and [7], which are built to collect and send data through a modem to a server. Although these are well-built systems that serve the purpose for a specific task, the user cannot interact with the system. Another unidirectional data transfer is presented in, which uses the Global System for Mobile Communications (GSM): a popular wireless choice for connectivity between the data-acquisition units and clients.

There are also several systems that allow data to be remotely accessed. As a solution to wireless data collection through the Internet, General Packet Radio Service (GPRS) is a popular choice in several applications. A surveillance system based on GPRS is presented in. A recent work [10] has presented a GPRS solution to the data-acquisition problem for remote areas. A distributed system capable of road vehicle locating, monitoring and telemetering with GPRS is presented in [11]. A long-distance data-collection system for the Earth tide gravimeter, collecting information on temperature, humidity, atmospheric pressure, etc., is designed with GPRS using a hard-coded static Internet Protocol (IP) address. These systems use GPRS without concerns about minimizing the cost of data transfer.

Similar types of Internet-based systems, such as those in, are designed to gather a bulk of data before serving them upon request. In these applications, data are compiled in a central server and are then served to the clients via the Internet. The client framework is in a central server and has all the applications. A person that needs to access any data must first access the server. An indirect access to the data-acquisition unit makes the system unattractive for real-time control applications, where direct interaction with the system may be required. The need to maintain an additional server will also

Corrseponding Author: S. Philomina, Department of ECE, Bharath University, Chennai, India.

increase the setup costs and the costs to maintain the acquisition systems, such as regular maintenance costs, system updates, etc.

Therefore, the central server has to be eliminated for a real-time system. The closest to this idea is published in. In this system, a reliable bidirectional Point-to-Point Protocol (PPP) link for real-time control and surveillance via a GSM network is formed. However, there is still no effort to minimize the operational costs (including the costs to transfer a large amount of data). In addition, this system is based on an industrial PC, thus making it an expensive solution. Interaction with the embedded unit is also an important issue.

In, an embedded PC card placed on the Internet allows limited interaction through commands sent through Transmission Control Protocol/IP (TCP/IP) and User Datagram Protocol.

In this paper, we propose a GPRS-based portable low-cost data-acquisition system, which can establish a reliable bidirectional connection for data-acquisition. The proposed system uniquely reduces the costs occurring from frequently requested data and eliminates the need for a well-established server. The system uses a dummy server for static information, thus optimizing the transfer of large data. The user can directly log in and interact with the embedded device in real time without the need to maintain an additional server. The system is modularly built, allowing different modules to be added. In addition, it is flexible to accommodate a wide range of measurement devices with appropriate interfaces.

In Section II, we will introduce the details of the afore mentioned system. In Section III, we will provide an example that will introduce some of the capabilities of the system using a collection of modules. In this paper, a camera, a Global Positioning System (GPS) and a temperature chip are connected to the embedded system to demonstrate its operation. The operational features are presented, which are needed to be considered for practical applications. Section IV presents the conclusion.

Interactive Data Acquisition System: The general principles of Internet-based control systems have been modelled in. Interactive Internet-based systems provide a way to monitor and adjust using standard web browsers and a PC. The target systems can be monitored and controlled independent from the location and the platform since standard web browsers can be used on the client side.

A typical data-acquisition system is made up of three components connected to each other via the Internet, as shown in Fig. 1. The data-acquisition system

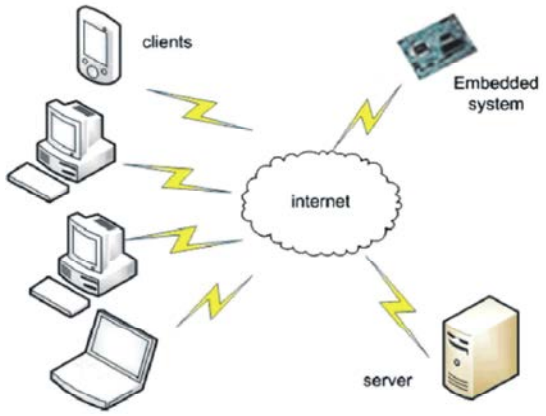


Fig. 1: Genral diagram of a data-acquisition and control system

needs to relay the acquired information to the requesting clients. The clients also need to send commands. If necessary, this is implemented through a server and then, an enormous amount of data transfer time would be consumed. Thus, alternative methods need to be explored.

Using the current based on eight MCU embedded gas measuring devices though to a certain extent to meet this demand flow testing, but can't adapt to increasingly complex monitoring system expansion in man-machine interface requirements, particularly demanding higher, networked degree deepen, management equipment under the trend of more intelligent, low microcontroller is ambition. Along with the development of the microelectronics technology, 16 and 32 bits embedded microprocessors gradually become the mainstream of the embedded system design. This kind of high end processor has completely can meet the small low energy intelligence equipment design requirements. The coal gas detection terminal equipment's main innovation point is realized with terminal testing equipment control operator of wireless communication, terminal equipment can follow the propulsion and convenient excavating of gas on cloth nets; points Terminal equipment can not only inform on-site staff of gas concentration status and can give PC industrial PC's site automatic control to provide real-time accurate control information, thus in the warning of the reliability of the real-time and control have greatly improved.

Qt ("cute", or unofficially as Q-T cue-tee) is a cross-platform application framework that is widely used for developing application software with a graphical user Interface (GUI) (in which cases Qt is classified as a widget toolkit) and also used for developing non-GUI programs such as command-line tools and consoles for servers.

Haavard Nord and Eirik Chambe-Eng (the original developers of Qt and the CEO and President, respectively, of Trolltech) began development of "Qt" in 1991.

Qt is free and open source software. All editions support many compilers, including the GCC C++ compiler and the Visual Studio suite.

On August 9, 2012, Digia acquired Qt software technologies from Nokia. About 125 Qt developers will be transferred to Digia, with the immediate goal of bringing Qt support to android, iOS and Windows 8 platforms.

Qt is most notably used in VLC media player, Safari Browser, Autodesk Maya, The Foundry's Nuke, Adobe Photoshop Elements, Skype, Virtual Box and Mathematica and by the European Space Agency, DreamWorks, Google, HP, KDE, Lucas film, Panasonic, Philips, Samsung, Siemens, Volvo, Walt Disney Animation Studios and Research In Motion. The Opera web browser also uses Qt, but only as an interface to the Linux platform.

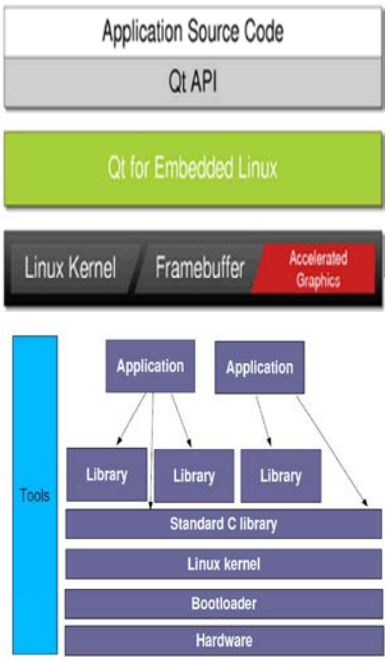
Embedded Linux is the use of Linux in embedded computer systems such as mobile phones, personal digital assistants, media players, set-top boxes and other consumer electronics devices, networking equipment, machine control, industrial automation, navigation equipment and medical instruments which have

- Low Memory Resources-RAM, Hard Disk
- Low Processing Power-CPU (400 MHz, <= 1GHz)
- Compact Size
- Low Cost Limitation
- Higher Performance and Efficiency in Speed, Power

Due to its low cost and ease of customization, Linux has been shipped in many consumer devices.

Linux is available for many architectures and an obvious candidate for an embedded system and it already is being used widely in this area. Its open nature makes it particularly attractive to developers. Development tool suites have begun to appear in response to the perceived need, although one can work without such luxury and employ less integrated tools already available in Linux. New embedded systems companies using Linux have opened for business and various older embedded systems companies have added Linux to their product line.

Qt for Embedded Linux is a C++ framework for GUI and application development for embedded devices. It runs on a variety of processors. Qt for Embedded Linux provides the standard Qt API for embedded devices with a lightweight Graphics system.



Qt is a cross-platform application and UI framework for writing web-enabled applications for desktop, mobile and embedded operating systems. This page contains links to articles and overviews explaining key components and techniques used in Qt development.

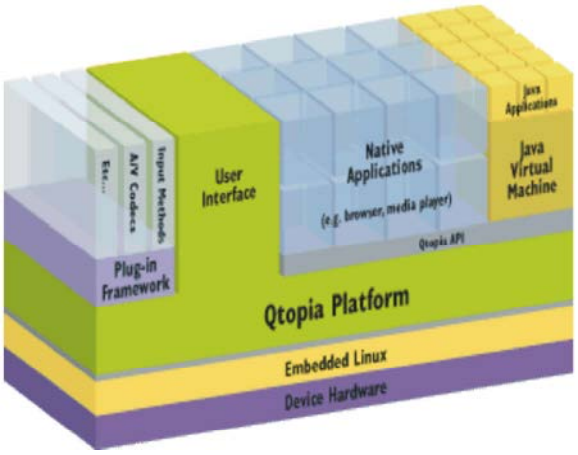
One of the most important issues with embedded systems is the need for a real-time operating system. The definition of real-time here varies quite a bit. To some people, real-time means responding to an event in the one-microsecond range, to others it is 50 milliseconds. The hardness of real-time also varies quite a bit. Some systems need hard real-time response, with short deterministic response latencies to events. However, on many systems, when analyzed closely, we see a response time requirement that is actually near real-time.

Often the real-time requirement is a tradeoff of time and buffer space. With memory getting cheaper and CPUs getting faster, near real-time is now more typical than hard real-time and many commercial operating systems that claim to be real-time are far from being hard real-time. Usually, when you get into the detailed design of these systems, there are warnings that the drivers' interrupts and applications must be very carefully designed in order to meet real-time requirements.

RT-Linux (Linux with real-time extensions) contains time critical functions to provide precise control over interrupt handling, through the use of an interrupt manager and does a good job of making sure that critical interrupts get executed when needed. The hardness

of this approach depends mostly on the CPU interrupt structure and context-switch hardware support. This approach is sufficient for a large range of real-time requirements. Even without the real-time extensions, Linux does pretty well at keeping up with multiple streams of events. For example, a Linux PC system on a low end Pentium is able to keep multiple 10BaseT interfaces executing effectively, while simultaneously running character-level serial ports at a full 56KBPS without losing any data.

Some real-time hardware and software Linux APIs to consider are RTLinux, RTAI, EL and Linux-SRT. RTLinux is a hard real-time Linux API originally developed at the New Mexico Institute of Technology. RTAI (DIAPM) is a spin-off of the RTLinux real-time API that was developed by programmers at the Department of Aerospace Engineering, Polytechnic Politecnico di Milano (DIAPM). EL/IX is a proposed POSIX-based hard real-time Linux API being promoted by Red Hat. And Linux-SRT is a soft real-time alternative to real-time APIs, which provides performance-enhancing capabilities to any Linux program without requiring that the program be modified or recompiled.



Qt is an application development framework aimed at enabling developers to compile binaries for Windows, Mac and Linux OSes from a single code base. Qt's API (application programming interface) comprises some 400 C++ class libraries, Trolltech says.

Trolltech claims its Qt API and development tools are "consistent across all supported platforms, enabling developers to learn one API and perform truly platform-independent application development and deployment." It allows development teams to create native applications for all major operating systems, from any of the supported platforms, the company adds.

Prior to today's release, Qt versions already existed for Windows desktops ("Qt/Windows"), OS X ("Qt/Mac"), Java ("QT/Jambi"), desktop Linux ("Qt/X11") and embedded Linux ("Qtopia"). Trolltech says QT/WinCE is a near-complete port of its API and developer tools to Microsoft's device-oriented Windows CE operating system.

More specifically, Qt/WinCE runs on Windows CE 5.0 and 6.0 and Windows Mobile 5.0 and 6.0. The only modules not included in the ported API are QtOpenGL and Qt3Support, according to the company. Trolltech also notes that developers wishing to access Windows Mobile-specific APIs, such as those for the PIM (personal information management) suite or telephony, must access these operating system features directly, bypassing Qt.

The MDK-ARM is a complete software development environment for Cortex™-M, Cortex-R4, ARM7™ and ARM9™ processor-based devices. MDK-ARM is specifically designed for microcontroller applications, it is easy to learn and use, yet powerful enough for the most demanding embedded applications.

- Complete support for Cortex-M, Cortex-R4, ARM7 and ARM9 devices
- Industry-leading ARM C/C++ Compilation Toolchain
- µVision4 IDE, debugger and simulation environment
- Keil RTX deterministic, small footprint real-time operating system (with source code)
- TCP Networking Suite offers multiple protocols and various applications
- USB Device and USB Host stacks are provided with standard driver classes
- ULINKpro enables on-the-fly analysis of running applications and records every executed Cortex-M instruction
- Complete Code Coverage information about your program's execution
- Execution Profiler and Performance Analyzer enable program optimization
- Numerous example projects help you quickly become familiar with MDK-ARM's powerful, built-in features.

- CMSIS Cortex Microcontoller Software Interface Standard compliant

GSM and GPRS are developed for cellular mobile communication. A GPRS connection with unlimited duration of connectivity is charged only for the data package transfers and adopted in several mobile remote control/access systems. GPRS becomes a cost-effective solution only if the data transfers can be optimized.

Once a GPRS connection has been established, queried data can be relayed to the client via a central server. Using a central server to relay the acquired data has some disadvantages. First, a central server needs a client interface framework. An additional data transfer corresponds to time delays before the data are made available to the client. In addition, since the server acts as a relay, no direct bidirectional communication between the client and the embedded system can be established. This makes the system unsuitable for real-time control applications. The basic idea behind real-time processing is that the embedded system is expected to respond to the queries in time. Real time should be fast enough in the context in which the system is operating and reliable as well. Real-time system correctness depends not only on the correctness of the logical result of the computation but also on the result delivery time. This method also increases the data transfer cost as the number of clients increases due to the access amount of data transfers via GPRS.

Direct communication, on the other hand, enables access to only relevant information in the embedded system by preprocessing the data. The embedded system should also handle the web services. This eliminates the need for a central server and reduces the amount of data sent from the remote unit since only the queried data will be transferred [12].

In the proposed system, the GPRS architecture and protocols are compliant with. This system is configured to be virtually online at all times in a GSM network. An admin script is executed after the boot of the operating system, initiating the GPRS connection software module. A PPP connection is established by a GPRS modem that works at 900/1800/ 1900 MHz operating frequencies. A PPP daemon (PPPD) is used to manage the PPP network connections between the client and the embedded module. The PPPD is responsible for setting up the GPRS parameters, such as the connection speed and compression.

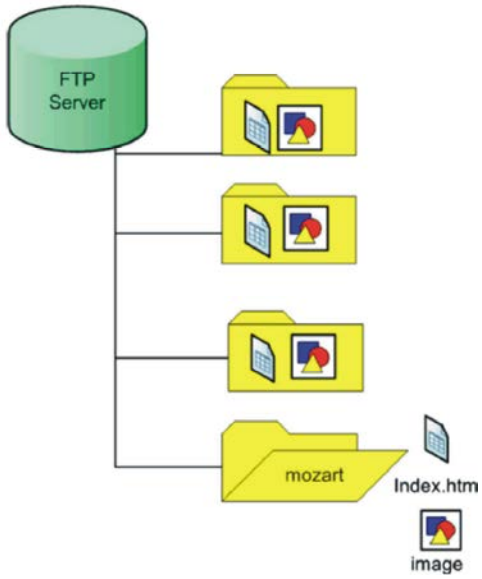


Fig. 2: Folder structure of tghe FTP server

To directly access an embedded system, the IP address of the embedded device should be made available to the client side. There are two choices available. A static (hard-coded) IP could be used, or the remote device should initiate a connection by reporting its IP. This choice is quite straightforward and simple. Although the usage cost remains unchanged, it requires a static IP setup by the service provider and involves monthly recurring costs. The static IP is preferred for its simplicity in designing a system; however, its overhead may be impractical. The other choice is to use a dynamic IP assigned through a Dynamic Host Configuration Protocol (DHCP) server of the GSM provider for every connection established. However, this IP needs to be known by any client requesting an access to the embedded server. One solution is to broadcast this IP to a dummy FTP server (where the bulky static information such as image data is also kept). The FTP server is a dummy server and does not require regular software updates or maintenance. The folder structure of the FTP server is shown in Fig. 2.

A script on the embedded device is configured to update its IP address on the FTP server in Hypertext Meta-Language as an index.htm file, under a folder uniquely named by its hostname. This script simply parses the current IP for that embedded device and sends an html file with the IP information of the embedded device to the FTP server. Once this file is in place, a direct connection can be established with the desired embedded device by a simple query. An example embedded system,

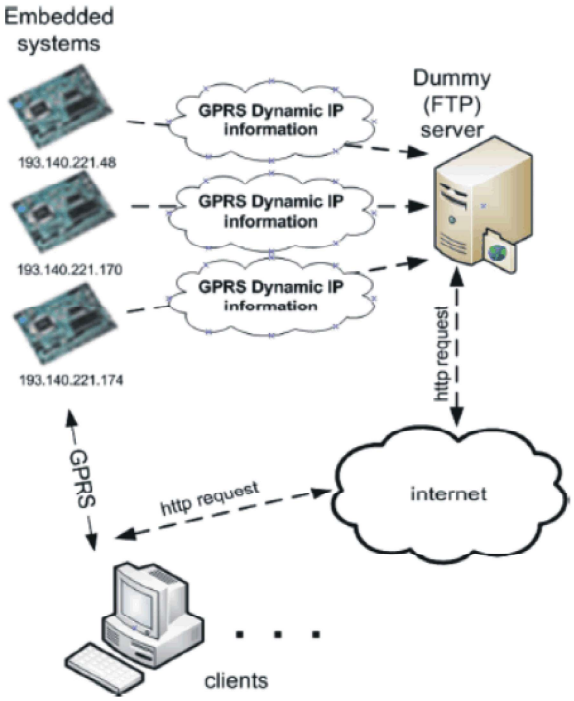


Fig. 3: IP address lookup for the stored IP

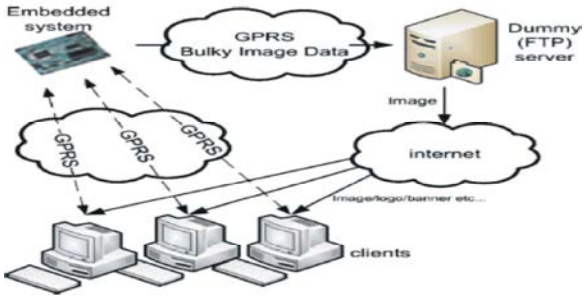


Fig. 4: Data management in the propsed system

named mozart (Fig. 2), can be queried from the FTP server by a simple command. The web browser processes the (index.htm) file in the specified folder as default; therefore, a file name is not needed for referencing.

This process is illustrated in Fig. 3. The DHCP approach is more flexible and works better compared with the static approach as a cost-effective solution, despite the necessity for a script running on the embedded server, one-time broadcasting its IP to the FTP server. The hypertext file placed on the FTP server by the embedded system and queried by the client is shown in Fig. 4. With this mechanism in place, the embedded system updates it IP information on the FTP server upon every reboot, which causes an IP refresh from the GSM service provider.

Data Management in the System: The Internet server is used to decrease the management costs by sending all the pictures (logo, picture, bar graphics, etc.) to the client through a server on the Internet. Text data such as coordinates, temperature and altitude are served from the embedded system. If bulky data are going to be sent, the embedded module is set to send the image only once via GPRS and placed on an FTP server. This approach eliminates the transfer of large data through GPRS more than once, thus reducing the transfer costs, particularly if more than one client is involved or multiple requests to the same data are needed, as shown in Fig. 5. A user interface, which is brought up upon establishing a direct connection, has links to the Common Gateway Interface (CGI) and Bourne Again Shell (BASH) script files executed on the embedded system. The code is compiled into the CGI format to be installed in the embedded board through a cross-compiler platform. BASH scripts are directly triggered by the applications.

Software and Operating System Choice: The Linux 2.4 kernel series with TCP/IP stack included has been chosen as the operating system for the embedded board. Only the bare minimum is installed, including the basics such as console try, serial ports, kernel side of the PPPD and support for memory and math emulation. The running kernel is around 1 MB of code built into a Flash memory. A scaled-down version of Linux has been used to reduce the memory footprint and the complexity.

The software running on the embedded system at the highest level is named the manager code, which will be explained in Section III with a sample implementation. In the design, the manager code controls the execution of other applications and is triggered once all the components of the operating system are up and running. The periodic operations and routine tasks are organized by a manager code. If a new data-access application is considered in future developments, its program can easily be added to the manager code as a periodic operation.

CONCLUSION

In this application, a low-cost, Internet-based data-acquisition and control system has been designed and implemented that should find interest from researchers. The Compared with other applications, this system has advantages in terms of allowing direct bidirectional communication and reducing overhead, which can be

vitally important for some real-time applications. The operational costs have been reduced by relinquishing the storage of large data to an FTP server on the Internet. The system is designed to support both static and dynamic IPs. A method to distribute the IP information has been developed. This cost-minimization effort is a big concern for mobile systems using wireless communication methods and has not been discussed before.

The overall cost advantage of the system in terms of the components used makes it an attractive choice for data-acquisition applications. The power demand of the device is still in the process of being improved by putting the attached devices into sleep mode at times when they are not in use to conserve power.

REFERENCES

1. Cypress Semiconductor Corporation, 3.3V 64K×16 Dual Port Static RAM CY7C028V Data Sheet. http://www.cypress.com, 2004, pp: 1-2, 4-6, 9-15.
2. Motorola Inc., Power PC™ MPC823e Reference Manual, http://www.motorola.com. 2000, pp: 63-73, 77-88, 219-270, 359-454, 967-997.
3. Cypress Semiconductor Corporation, Interfacing the CY7C028V with Motorola's MPC860 Power QUICC™ Processor. http://www.cypress. com, 2002, pp: 1-4.
4. Ling-li Dai and Jin Ou Yang, 2002. Linux Kernel Analysis and Application Examples. National Defense Industry Press: Beijing, pp: 82-84, 88-90, 192-195, 203-208.
5. Rubini, A. and J. Corbet, 2002. Translated by Yong-ming Wei. Linux Device Driver. China Electric Power Press: Beijing, pp: 59-63, 66-76, 80-86, 118-128, 204-209.
6. Zhao-ling Q.I., 2004. Embedded Linux Based on PowerPC. Beijing University of Aeronautics and Astronautics Press: Beijing, pp: 123-137, 142-145, 180-182, 190-197.
7. De-cao Mao and Xi-ming Hu Linux Kernel, 2001. Scenarios and Analyses. Zhejiang University Press: Hangzhou, pp: 127-177, 326-329.
8. Shafaq Sherazi and Habib Ahmad, 2014. Volatility of Stock Market and Capital Flow Middle-East Journal of Scientific Research, 19(5): 688-692.
9. Kishwar Sultana, Najm ul Hassan Khan and Khadija Shahid, 2013. Efficient Solvent Free Synthesis and X Ray Crystal Structure of Some Cyclic Moieties Containing N-Aryl Imide and Amide, Middle-East Journal of Scientific Research, 18(4): 438-443.
10. Pattanayak, Monalisa and P.L. Nayak, 2013. Green Synthesis of Gold Nanoparticles Using Elettaria cardamomum (ELAICHI) Aqueous Extract World Journal of Nano Science & Technology, 2(1): 01-05.
11. Chahataray, Rajashree and P.L. Nayak, 2013. Synthesis and Characterization of Conducting Polymers Multi Walled Carbon Nanotube-Chitosan Composites Coupled with Poly (P-Aminophenol) World Journal of Nano Science & Technology, 2(1): 18-25.
12. Parida, Umesh Kumar, S.K. Biswal, P.L. Nayak and B.K. Bindhani, 2013. Gold Nano Particles for Biomedical Applications World Journal of Nano Science & Technology, 2(1): 47-57.